

AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

- 1 1. (Original) A method comprising:
2 storing first tuples in a first table in a database system;
3 storing second tuples in a second table in the database system;
4 partitioning the first and second tuples into plural portions;
5 redistributing the first and second tuples to plural nodes according to the
6 partitioning; and
7 hash joining the first and second tuples to produce result tuples as the first and
8 second tuples are being redistributed to the plural nodes.
- 1 2. (Cancelled)
- 1 3. (Original) The method of claim 1, further comprising:
2 retrieving the result tuples at random.
- 1 4. (Original) The method of claim 1, hash joining the first and second tuples to
2 produce result tuples as the first and second tuples are being redistributed to the plural nodes
3 further comprising:
4 producing result tuples at one of the plural nodes; and
5 simultaneously producing result tuples at a second of the plural nodes.
- 1 5. (Original) The method of claim 1, wherein redistributing the first and second
2 tuples to plural nodes comprises redistributing based on split vectors containing predefined
3 ranges.
- 1 6. (Original) The method of claim 5, wherein partitioning the first and second tuples
2 into plural portions comprises:
3 partitioning first and second tuples into hash tables in each node.

1 7. (Original) The method of claim 6, wherein hash joining the first and second tuples
2 comprises:
3 allocating a portion of a memory to a first hash table;
4 allocating a second portion of the memory to a second hash table; and
5 hash joining first tuples in the first hash table with second tuples in the second
6 hash table.

1 8. (Original) The method of claim 7, wherein hash joining the first and second tuples
2 comprises:
3 determining that the portion of the memory allocated to the first hash table is full;
4 allocating a stable storage to the first hash table; and
5 storing first tuples in the stable storage.

1 9. (Original) The method of claim 8, further comprising:
2 continuing to store second tuples in the second hash table; and
3 hash joining second tuples in the second hash table with first tuples in the first
4 hash table.

1 10. (Original) The method of claim 9, further comprising:
2 determining that the second portion of the memory allocated to the second hash
3 table is full;
4 allocating a second stable storage to the second hash table;
5 storing second tuples in the second stable storage; and
6 hash joining second tuples in the second stable storage with first tuples in the first
7 hash table.

1 11. (Original) The method of claim 10, wherein hash joining the first and second
2 tuples comprises:
3 generating a third hash table once all first tuples and second tuples are
4 redistributed to each node;
5 retrieving one of the first tuples from the stable storage;
6 hash joining the one of the first tuples with tuples in the second hash table; and
7 storing the one of the first tuples in the third hash table.

1 12. (Original) The method of claim 11, further comprising:
2 retrieving one of the second tuples from the second stable storage; and
3 hash joining the one of the second tuples with tuples in the third hash table.

1 13. (Previously Presented) A database system comprising:
2 a plurality of nodes; and
3 instructions for enabling the database system to:
4 store first tuples in a first table distributed across the plurality of
5 nodes;
6 store second tuples in a second table distributed across the plurality
7 of nodes;
8 partition the first and second tuples into plural portions;
9 redistribute the first and second tuples to the plurality of nodes according
10 to the partitioning; and
11 hash join the first and second tuples to produce result tuples as the first
12 and second tuples are being redistributed to the plurality of nodes.

1 14. (Cancelled)

1 15. (Previously Presented) The database system of claim 13, wherein the result tuples
2 are available at random.

1 16. (Previously Presented) The database system of claim 13, wherein each node
2 comprises a memory, and wherein the instructions further partition the first and second tuples
3 into plural portions by:
4 partitioning first tuples into first hash tables; and
5 partitioning second tuples into second hash tables, wherein the hash tables are in
6 the memory.

1 17. (Previously Presented) The database system of claim 16, wherein the instructions
2 further:
3 allocate a portion of the memory to the first hash table;
4 allocate a second portion of the memory to the second hash table; and
5 hash join first tuples in the first hash table with second tuples in the second hash
6 table.

1 18. (Previously Presented) The database system of claim 17, wherein the instructions
2 further:
3 determine that the portion of the memory allocated to the first hash table is full;
4 and
5 store first tuples in a stable storage.

1 19. (Previously Presented) The database system of claim 18, wherein the instructions
2 further:
3 continue to store second tuples in the second hash table; and
4 hash join second tuples in the second hash table with first tuples in the first hash
5 table.

1 20. (Previously Presented) The database system of claim 19, wherein the instructions
2 further:
3 determine that the second portion of the memory allocated to the second hash
4 table is full;
5 allocate a second stable storage to the second hash table;
6 store second tuples in the second stable storage; and
7 hash join second tuples in the second stable storage with first tuples in the first
8 hash table.

1 21. (Previously Presented) The database system of claim 20, wherein the instructions
2 further:
3 generate a third hash table once all first tuples and second tuples are redistributed
4 to each node;
5 retrieve one of the first tuples from the stable storage;
6 hash join the one of the first tuples with tuples in the second hash table; and
7 store the one of the first tuples in the third hash table.

1 22. (Previously Presented) The database system of claim 21, wherein the instructions
2 further:
3 retrieve one of the second tuples from the second stable storage; and
4 hash join the one of the second tuples with tuples in the third hash table.

1 23. (Previously Presented) An article comprising a medium storing instructions for
2 enabling a processor-based system to:

3 store first tuples in a first table in a database system;
4 store second tuples in a second table in the database system;
5 partition the first and second tuples into plural portions;
6 redistribute the first and second tuples to plural nodes of the database system
7 according to the partitioning; and
8 hash join the first and second tuples to produce result tuples as the first and
9 second tuples are being redistributed to the plural nodes.

1 24. (Original) The article of claim 23, further storing instructions for enabling a
2 processor-based system to:

3 retrieving the result tuples once the hash join is performed.

1 25. (Original) The article of claim 24, further storing instructions for enabling a
2 processor-based system to:

3 redistribute based on split vectors containing predefined ranges.

1 26. (Original) The article of claim 25, further storing instructions for enabling a
2 processor-based system to:

3 partition first and second tuples into hash tables in each node.

1 27. (Original) The article of claim 26, further storing instructions for enabling a
2 processor-based system to:

3 allocate a portion of a memory to a first hash table;
4 allocate a second portion of the memory to a second hash table; and
5 hash join first tuples in the first hash table with second tuples in the second hash
6 table.

1 28. (Original) The article of claim 27, further storing instructions for enabling a
2 processor-based system to:
3 determine that the portion of the memory allocated to the first hash table is full;
4 and
5 store first tuples in a stable storage.

1 29. (Original) The article of claim 28, further storing instructions for enabling a
2 processor-based system to:
3 continue to store second tuples in the second hash table; and
4 hash join second tuples in the second hash table with first tuples in the first hash
5 table.

1 30. (Original) The article of claim 29, further storing instructions for enabling a
2 processor-based system to:
3 determine that the second portion of the memory allocated to the second hash
4 table is full;
5 allocate a second stable storage to the second hash table;
6 store second tuples in the second stable storage; and
7 hash join second tuples in the second stable storage with first tuples in the first
8 hash table.

1 31. (Original) The article of claim 30, further storing instructions for enabling a
2 processor-based system to:
3 generate a third hash table once all first tuples and second tuples are redistributed
4 to each node;
5 retrieve one of the first tuples from the stable storage;
6 hash join the one of the first tuples with tuples in the second hash table; and
7 store the one of the first tuples in the third hash table.

1 32. (Original) The article of claim 31, further storing instructions for enabling a
2 processor-based system to:

3 retrieve one of the second tuples from the second stable storage; and
4 hash join the one of the second tuples with tuples in the third hash table.

1 33. (Previously Presented) The method of claim 1, wherein storing the first tuples in
2 the first table comprises distributing the first tuples across the plural nodes of the database
3 system, and wherein storing the second tuples in the second table comprises distributing the
4 tuples across the plural nodes.

1 34. (Previously Presented) The method of claim 33, wherein redistributing the first
2 and second tuples comprises redistributing the first and second tuples to the plural nodes of the
3 database system.

1 35. (Previously Presented) The article of claim 23, wherein storing the first tuples in
2 the first table comprises storing the first tuples in the first table distributed across the plural
3 nodes of the database system, and wherein storing the second tuples in the second table
4 comprises storing the second tuples distributed across the plural nodes of the database system.

1 36. (New) The method of claim 1, wherein each of the nodes contains a first hash
2 table to receive first tuples, and a second hash table to receive second tuples, the method further
3 comprising:

4 storing redistributed first tuples in respective first hash tables; and
5 storing redistributed second tuples in respective second hash tables.

1 37. (New) The method of claim 36, wherein hash joining first tuples and second
2 tuples comprises hash joining first tuples and second tuples from corresponding first and second
3 hash tables.

1 38. (New) The database system of claim 13, wherein each of the nodes contains a first
2 hash table to receive first tuples, and a second hash table to receive second tuples,
3 wherein the instructions further:
4 store redistributed first tuples in respective first hash tables; and
5 store redistributed second tuples in respective second hash tables.

1 39. (New) The database system of claim 38, wherein the instructions further hash join
2 the first tuples and the second tuples from corresponding first and second hash tables.

1 40. (New) The article of claim 23, wherein each of the nodes contain a first hash table
2 to receive first tuples, and a second hash table to receive second tuples, wherein the instructions
3 when executed cause the processor-based system to further:
4 store redistributed first tuples in respective first hash tables; and
5 store redistributed second tuples in respective second hash tables.

1 41. (New) The article of claim 40, wherein hash joining first tuples and second tuples
2 comprises hash joining first tuples and second tuples from corresponding first and second hash
3 tables.